IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received July 15, 2019, accepted August 19, 2019, date of publication September 13, 2019, date of current version September 24, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2940983

Adaptive FH-SVM for Imbalanced Classification

QI WANG[1,2,3], YINGJIE TIAN[2,3,4], AND DALIAN LIU[5]
[1]School of Mathematical Sciences, University of Chinese Academy of Science, Beijing 100049, China
[2]Research Center on Fictitious Economy and Data Science, Chinese Academy of Sciences, Beijing 100190, China
[3]Key Laboratory of Big Data Mining and Knowledge Management, Chinese Academy of Sciences, Beijing 100190, China
[4]School of Economics and Management, University of Chinese Academy of Sciences, Beijing 100190, China
[5]Department of Basic Course Teaching, Beijing Union University, Beijing 100101, China

Corresponding author: Dalian Liu (ldtdalian@buu.edu.cn)

This work was supported in part by the Science and Technology Service Network Program of Chinese Academy of Sciences (STS Program), under Grant KFJ-STS-ZDTP-060, in part by the National Natural Science Foundation of China under Grant 71731009, Grant 61472390, Grant 71331005, and Grant 91546201, and in part by the Premium Funding Project for Academic Human Resources Development in Beijing Union University.

ABSTRACT Support vector machines (SVMs), powerful learning methods, have been popular among machine learning researches due to their strong performance on both classification and regression problems. However, traditional SVM making use of Hinge Loss cannot deal with class imbalance problems, because it applies the same weight of loss to each class. Recently, Focal Loss has been widely used for deep learning to address the imbalanced datasets. The significant effectiveness of Focal loss attracts the attention in many fields, such as object detection, semantic segmentation. Inspired by Focal loss, we reconstructed Hinge Loss with the scaling factor of Focal loss, called FH Loss, which not only deals with the class imbalance problems but also preserve the distinctive property of Hinge loss. Owing to the difficulty of the trade-off between positive and negative accuracy in imbalanced classification, FH loss pays more attention on minority class and misclassified instances to improve the accuracy of each class, further to reduce the influence of imbalance. In addition, due to the difficulty of solving SVM with FH loss, we propose an improved model with modified FH loss, called Adaptive FH-SVM. The algorithm solves the optimization problem iteratively and adaptively updates the FH loss of each instance. Experimental results on 31 binary imbalanced datasets demonstrate the effectiveness of our proposed method.

INDEX TERMS Focal loss, hinge loss, class imbalance, support vector machines (SVMs).

I. INTRODUCTION

Class imbalance problem has always been a challenging issue in traditional machine learning and deep learning. It appears when the number of instances in some classes is significantly more than that in others. In the binary classification problem, the class containing more instances is called the majority class whereas the other one is called the minority class. It's worth noting that we mainly focus on the binary classification problem in this paper where the majority class and the minority class are referred to as 'negative' and 'positive' respectively, and the proposed algorithm can be extended and applied to multiclass datasets according to one-versus-all scheme. In practice, imbalanced datasets are frequently occurred in various domains, such as object detection, medical diagnosis [1], [2], text classification [3], credit risk assessment [4] and so on.

The associate editor coordinating the review of this manuscript and approving it for publication was Victor S. Sheng.

In general, the objective of learning methods is to obtain high accuracy of the whole dateset by optimizing target function. In the imbalanced dataset, a high accuracy can be obtained by the way of classifying all instances as majority class, while all minority class instances are misclassified. In particular, the classifier even loses the ability of classification for extreme class imbalanced datasets, even if the accuracy is very high. In terms of imbalanced datasets, it's more significant to concentrate on the performance of each individual class other than the whole dateset. For example, in cancer detection, we should pay more attention to identify cancer patients as accurately as possible. As a consequence, many conventional classifiers can't maintain good performance any more in imbalanced datesets, such as Support Vector Machine (SVM) [5].

Support Vector Machines (SVMs), a powerful learning method, have been popular among machine learning researches due to its strong performance on both classification and regression problems. However, the SVM classifier trained on the imbalanced dataset tends to be biased to the majority class and a suboptimal model comes about [6], [7], which doesn't work as expected. In order to make up this deficiency, there have been amount of techniques proposed. These methods can be summarized as external methods and internal methods [8].

External methods contain two categories: data preprocessing methods and ensemble learning methods. The former tends to balance the skewed class distribution by the means of over-sampling to the minority class or under-sampling to the majority class, which is applied before training SVM models. The main sampling methods include randomly under-sampling (RUS), randomly over-sampling (ROS) and Synthetic Minority Over-sampling Technique (SMOTE) [9]. Later on, various sampling methods occurred, such as MSMOTE [10], MWMOTE [11], etc. The latter, i.e. ensemble learning, aims at dividing the majority class into multiple subdatasets, each of which has the same number of instances as the minority class, then developing different SVM classifiers with the same original minority class and different majority class subdatasets, finally merging classifiers following a certain criterion. The combination of ensemble approach and data preprocessing method is commonly employed. The representative algorithms include SMOTEBoost [12], RUSBoost [13], EasyEnsemble [14].

Internal methods refer to the algorithmic modifications to the standard SVM. One of the classical modifications is cost-sensitive learning, which first appeared in Different Error Costs (DEC) method [15] with respect to SVM. DEC method modifies the same misclassification cost for each class to two different costs for positive and negative classes respectively. Generally speaking, the ratio of two misclassification costs is given in advance, which should adhere to the rule of a higher positive cost than the negative one. Inspired by DEC, zSVM [16] adjusting weights in the decision function and FSVM-CIL [17] combining FSVM [18] with DEC were yielded. In addition, one class classification [19] is also an interesting algorithmic modification method, which converts the class imbalance problem into an anomaly detection problem. One class SVM (OC-SVM) [20] estimates a SVM classifier with only one class instances. Later on, many improved algorithms [21], [22] appeared on the basis of OC-SVM. In practice, there are several methods combining external and internal approaches, some of which have better performance than the utilization of either of these ones alone [6].

However, there is a worth stressing point to design algorithms solving the class imbalance problem. Traditionally, our starting point is to expect positive instances to receive the same or more attention as negative ones, due to less quantity of positive class. The less attention, the worse the solution to class imbalance. But if the attention is too high, the model will be biased towards positive class, in which the increase of positive classification accuracy is at the cost of decreasing negative classification accuracy. In practice, each class's accuracy is yet crucial to imbalanced datasets. For example, in crack detection [33], [34], we hope that the final detected cracks are neither under-reported (false negatives) nor misreported (false positives). Nevertheless, the trade-off between positive and negative accuracy is not only important but also difficult to master.

In the exiting methods, most only concentrate on positive accuracy. In order to better balance positive and negative accuracies, a novel loss function called FH Loss is proposed in this paper. The loss function is a dynamically scaled hinge loss, in which the scaling factor focuses on improving the weight of either minority class instances or misclassified instances. That is to say, the loss of the misclassified negative instances are not much lower than positive instances so that the negative class accuracy is improved. In order to solve easily, the algorithm named Adaptive FH-SVM is given which can achieve the adaptive effect by updating each instance loss per iteration. The experiments carried out on 31 benchmark datasets show the effectiveness of our proposed FH Loss and Adaptive FH-SVM compared with previous state-of-the-art techniques.

The remainder of this paper is shown as below. Section 2 gives a brief summary on related work. Section 3 focuses on the proposed loss function and relative algorithm. In section 4, several experiments are conducted to verify the effectiveness of the proposed method. The conclusions are given in section 5.

II. RELATED WORK

In this section, we briefly introduce some related works about the proposed FH loss function. Specifically, it is preliminaries of hinge loss, weighted hinge loss and focal loss that this section has something to do with.

A. SVM WITH HINGE LOSS

SVMs have been widely applied in various fields since Cortes and Vapnik [5] proposed. The main idea of SVM is to find a separating hyperplane maximizing the margin between two classes as possible as it can.

Given a training set

$$S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\} \tag{1}$$

where $x_i \in \mathcal{X} \subseteq R^m$, $y_i \in \mathcal{Y} = \{-1, 1\}$, $i = 1, \ldots, N$, the standard SVM can be represented as a constrained convex quadratic programming problem (QPP), as shown below.

$$\begin{aligned} &\min_{w,b,\eta} \frac{1}{2} w^\top w + C \sum_{i=1}^{N} \eta_i \\ &s.t.\ y_i(w^\top \phi(x_i) + b) \geq 1 - \eta_i, \quad i = 1, \ldots, N \\ &\quad \eta_i \geq 0, \quad i = 1, \ldots, N \end{aligned} \tag{2}$$

where $\eta = (\eta_1, \ldots, \eta_N)^\top$ is the slack variable; $C > 0$ is a pre-given hyper-parameter to balance the margin and the loss; $\phi(\cdot)$ maps samples into a higher-dimensional space, defined implicitly by a relevant kernel function [25] $\kappa(x, y) = \langle\phi(x), \phi(y)\rangle$; w is the weight vector whose dimension is the same as the feature space's and $b \in R$ is the bias term.

By solving the primal problem (2) or its dual problem, the decision hyperplane

$$f(x) = w^\top \phi(x) + b = 0 \tag{3}$$

can be obtained.

From the perspective of loss function, the optimization problem above (2) is equivalent to a unconstraint problem with Hinge Loss function

$$L_{hinge}(s - z) = H_s(z) = \max\{0, s - z\} \tag{4}$$

where s is the Hinge point. The unconstraint program can be formulated as:

$$\min_{w,b} \frac{1}{2} w^\top w + C \sum_{i=1}^{N} L_{hinge}(1 - y_i(w^\top \phi(x_i) + b)) \tag{5}$$

Obviously, it follows the form of structural risk minimization (SRM) [23], where the first item is regularizer suggesting model complexity and the latter indicates the total empirical loss. Making use of hinge loss induces the loss of merely fully-properly classified instances is zero. This property guarantees good performance of classic SVM for balanced datasets.

For notational convenience, the hinge loss SVM is short for C-SVM in the following.

B. WEIGHTED SVM

Due to the target of total accuracy, the C-SVM doesn't have good performance any more in imbalanced datasets. For the purpose of dealing with class imbalance problem, DEC [15] occurred, called weighted SVM in this paper. The modified optimization is stated as follows:

$$\begin{aligned} &\min_{w,b,\eta} \frac{1}{2} w^\top w + C(C_+ \sum_{i:y_i=1} \eta_i + C_- \sum_{i:y_i=-1} \eta_i) \\ &s.t.\ y_i(w^\top \phi(x_i) + b) \geq 1 - \eta_i, \quad i = 1, \ldots, N \\ &\quad \eta_i \geq 0, \quad i = 1, \ldots, N \end{aligned} \tag{6}$$

where C_+ and C_- are the cost of misclassifying positive and negative class respectively, which satisfy commonly $C_+ > C_-$ because of the skewed class distribution.

It is critical to choose the ratio of C_+ to C_-. If the ratio is slightly more than 1, there is still the effect of data imbalance. However, if the ratio is too large, the decision boundary will bias towards the positive. In many algorithms, it is defined as the ratio of the number of negative samples to that of positive samples [6]. In this situation, since the cost of each instance in the same class is identical and fixed, the classifying result often can be improved. In other algorithms, two costs C_+, C_- are determined by cross-validation, in which two circumstances mentioned above will appear. The solution is to modify weights by some means to balance two situations.

C. FOCAL LOSS

In order to improve the accuracy of dense object detection, Tsung-Yi et al. proposed the Focal Loss recently [24]. This loss function is developed based on the standard cross entropy loss function [35] for binary classification

$$L_{CE}(\tilde{p}) = -\log(\tilde{p}) \tag{7}$$

where

$$\tilde{p} = \begin{cases} p, & if\ y = 1 \\ 1 - p, & otherwise \end{cases} \tag{8}$$

and $p \in [0, 1]$ is the probability that the sample is predicted as class 1, i.e. $p = P(y = 1|x)$.

To enable the cross entropy to address class imbalance, a balancing factor $\tilde{\tau} \in [0, 1]$ is introduced, defined analogously to $\tilde{p}$. That is to say,

$$\tilde{\tau} = \begin{cases} \tau, & if\ y = 1 \\ 1 - \tau, & otherwise \end{cases} \tag{9}$$

Therefore, the modified cross entropy loss is:

$$L_{\tau-CE}(\tilde{p}) = -\tilde{\tau} \log(\tilde{p}) \tag{10}$$

As experiments show in [24], the large number of well-classified instances overwhelm the entire loss while $\tau-$balanced the cross entropy loss function is applied during training. As we know, the instance containing more information is not easy-classified one but hard-classified one. So [24] adds a modulating factor $(1 - \tilde{p})^\gamma$ to $\tau-$balanced cross entropy loss, in order to force the model to focus on samples with high probability of being misclassified. Thus, Focal Loss is defined as:

$$\begin{aligned} L_{focal}(\tilde{p}) &= (1 - \tilde{p})^\gamma L_{\tau-CE}(\tilde{p}) \\ &= -\tilde{\tau}(1 - \tilde{p})^\gamma \log(\tilde{p}) \end{aligned} \tag{11}$$

where the focusing factor $\gamma \geq 0$ is a hyper-parameter.

III. OUR APPROACH

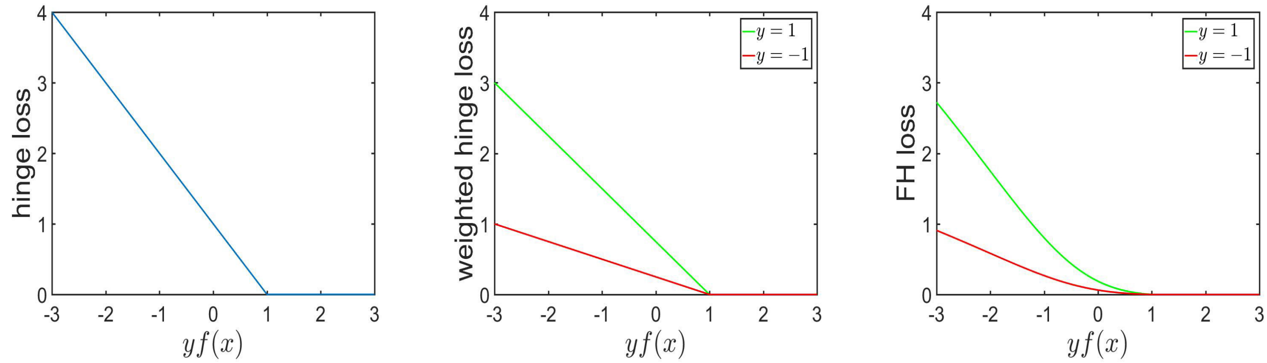
In this section, we present our proposed loss function, FH Loss, in detail. FH Loss not only catches the sensitivity of imbalance ratio, but also guarantees the accuracy of each class.

A. FH LOSS

As introduced in section I, the key to tackle the class imbalance problem is to improve the positive and negative class accuracy, that is, to reduce false positives and false negatives at the same time. However, most algorithms only focus on the positive accuracy during training and finally evaluate the model by the balanced index (e.g. *G-mean* [30], *F-measure* [31]), which are not suitable enough.

In order to improve the accuracy of each class effectively, we reshape Hinge loss by a scaling factor, inspired by Focal Loss. So the novel loss is named FH Loss and its form is stated as:

$$L_{\mathrm{FH}}(y, f(x)) = \max\left\{0, \tilde{\tau}\left(1 - \tilde{p}\right)^\gamma \left(1 - yf(x)\right)\right\} \tag{12}$$



(a) hinge loss (b) weighted hinge loss (c) FH loss

FIGURE 1. The illustration of (a)hinge loss: $L(x) = \max\{0, 1 - yf(x)\}$, (b)weighted hinge loss: $L(x) = \max\{0, \tilde{\tau}(1 - yf(x))\}$ and (c)FH loss: $L(x) = \max\left\{0, \tilde{\tau}(1-\tilde{p})^{\gamma}(1-yf(x))\right\}$. The representations of $\tilde{\tau}, \tilde{p}$ are shown in equations (13)(14). The settings of parameters in this figure are $\tau = 0.75$ and $\gamma = 2$. (a) hinge loss. (b) weighted hinge loss. (c) FH loss.

where

$$\tilde{\tau} = \begin{cases} \tau, & \text{if } y = 1 \\ 1-\tau, & \text{otherwise} \end{cases} \tag{13}$$

$$\tilde{p} = \begin{cases} p, & \text{if } y = 1 \\ 1-p, & \text{otherwise} \end{cases} \tag{14}$$

and $\tau \in [0, 1]$, $\gamma \geq 0$ as shown before, and sigmoid function

$$p = \sigma(f(x)) = \frac{1}{1+e^{-f(x)}} \tag{15}$$

transfers the output value into a probability belonging to [0,1]. Thus, the FH loss can be further written as:

$$L_{\mathrm{FH}}(y, f(x)) = \max\left\{0, \frac{\tilde{\tau}}{(1+e^{yf(x)})^{\gamma}}\left(1-yf(x)\right)\right\} \tag{16}$$

According to the definition of $\tilde{p}$, it indicates the probability that the instance is predicted correctly. If the instance is misclassified, the relative $\tilde{p}$ is small, further the factor $(1-\tilde{p})^{\gamma}$ is near 1. On the contrary, a well-classified instance gains a factor score near 0. Thus, the loss of misclassified instances in each class is up-weighted. For another thing, $\tilde{\tau}$ balances the loss between the positive class and negative class. All in all, FH Loss makes the SVM model pay more attention on both positive instances and misclassified ones and take on better performance for class imbalance.

To further analyze the proposed FH loss, we compare hinge loss, weighted hinge loss and FH loss in Figure 1. It can be seen from the figure that hinge loss allocates the same misclassified cost to each class such that it can't deal with the imbalanced dataset effectively. The introduction of $\tilde{\tau}$ in weighted hinge loss gives more attention on positive class, although the loss of each instance descends. Compared to weighted hinge loss, FH loss up-weights not only positive class by $\tilde{\tau}$ but also misclassified instances by the modulating factor $(1-\tilde{p})^{\gamma}$. In terms of each class, FH loss enlarges the loss contribution of misclassified samples and is beneficial to improve the accuracy of individual class. In terms of both classes, FH loss prevents that the increase of positive classification accuracy is at the cost of decreasing negative classification accuracy, to some extend. Because the loss contribution of negative support vectors increases than that in weighted hinge loss. For example, a negative sample of $f(x) = 2$ and a positive sample of $f(x) = 0$ have the same weighted hinge loss whereas the FH loss of the negative sample is nearly three times as much as the FH loss of the positive sample. Therefore, the FH loss is effective for imbalanced datasets.

In the next content, we will apply FH Loss to SVM and construct the relative optimization problem.

B. SVM WITH FH LOSS

Under this given training dataset described above, the framework of the primal optimization problem can be written as:

$$\min_{w,b} \frac{1}{2}w^{\top}w + C\sum_{i=1}^{N} L(x_i, y_i) \tag{17}$$

where L is an arbitrary loss function.

Therefore, apply our FH Loss (12) to the framework (17) and the unconstraint optimization is:

$$\begin{aligned} \min_{w,b} \frac{1}{2}w^{\top}w + C\sum_{i=1}^{N} \max\Big\{0, \\ \tilde{\tau}\left(1-\tilde{p_i}\right)^{\gamma}\left(1-y_i(w^{\top}\phi(x_i)+b)\right)\Big\} \end{aligned} \tag{18}$$

where

$$\tilde{p_i} = \begin{cases} p_i, & \text{if } y = 1 \\ 1-p_i, & \text{otherwise} \end{cases} \tag{19}$$

and

$$p_i = \sigma(f(x_i)) = \frac{1}{1+e^{-(w^{\top}\phi(x_i)+b)}} \tag{20}$$

for $i = 1, \ldots, N$.

For optimized convenience, we introduce the slack vector and apply (19)(20) to (18), then get the constrained optimization:

$$\begin{aligned}&\min_{w,b,\eta}\ \frac{1}{2}w^\top w + C\sum_{i=1}^{N}\eta_i\\&s.t.\ y_i(w^\top\phi(x_i)+b) \geq 1 - \frac{(1+e^{y_i(w^\top\phi(x_i)+b)})^\gamma}{\tilde{\tau}}\eta_i\\&\qquad \eta_i \geq 0,\ for\ all\ i=1,\ldots,N\end{aligned} \tag{21}$$

It's obvious that the target function of (21) is quadratic and the first kind of constraint is no longer linear, due to the existing of sigmoid operation in FH loss. Thus, there is a certain difficulty of solving the optimization problem (21). To address this issue, in the next section, we search for the modified form of FH loss and the corresponding algorithm, in which the final decision boundary is obtained iteratively and the subproblem is a constrained convex quadratic programming in each iteration.

C. MODIFIED FH LOSS

In this section, we modify FH loss by altering the sigmoid term $\tilde{p}$ to a known information instead of the item to be optimized. The algorithm SVM with FH loss is amended to be iterative mode accordingly, named Adaptive FH-SVM because of automatical adjustment of the loss.

The modification of FH Loss, called $\hat{\text{FH}}$ Loss, is stated as follows, which is applied to algorithm and experiments in practice:

$$L_{\hat{\text{FH}}}(y, f(x)) = \max\left\{0, \tilde{\tau}\left(1-\tilde{p}\right)^\gamma\left(1-yf(x)\right)\right\} \tag{22}$$

Here $\tilde{p}$ in (22) is a prior information or a posteriori information, namely a constant, while $\tilde{p}$ in (12) is an unknown item, namely a variable.

Apply the $\hat{\text{FH}}$ Loss (22) to the framework (17) and introduce the slack variable, then the $\hat{\text{FH}}$-SVM primal problem is:

$$\begin{aligned}&\min_{w,b,\eta}\ \frac{1}{2}w^\top w + C\sum_{i=1}^{N}\eta_i\\&s.t.\ y_i(w^\top\phi(x_i)+b) \geq 1 - \frac{1}{\tilde{\tau}(1-\tilde{p}_i)^\gamma}\eta_i\\&\qquad \eta_i \geq 0,\ for\ all\ i=1,\ldots,N\end{aligned} \tag{23}$$

This is a convex quadratic programming, so we can get its dual problem. First of all, we introduce the following Lagrangian function:

$$\begin{aligned}L(w,b,\eta,\alpha,\beta) = \frac{1}{2}w^\top w + C\sum_{i=1}^{N}\eta_i - \sum_{i=1}^{N}\beta_i\eta_i + \sum_{i=1}^{N}\alpha_i\\ \left(1-\frac{1}{\tilde{\tau}(1-\tilde{p}_i)^\gamma}\eta_i - y_i\left(w^\top\phi(x_i)+b\right)\right)\end{aligned} \tag{24}$$

where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_N)^\top$ and $\beta = (\beta_1, \beta_2, \ldots, \beta_N)^\top$ are Lagrangian multiplier vectors. The Karush-Kuhn-Tucker (KKT) conditions for the problem (23) are obtained by:

$$\begin{aligned}\partial L/\partial w &= w - \sum_{i=1}^{N}\alpha_i y_i\phi(x_i) = 0\\ \partial L/\partial b &= \sum_{i=1}^{N}\alpha_i y_i = 0\\ \partial L/\partial \eta_i &= C - \frac{\alpha_i}{\tilde{\tau}(1-\tilde{p}_i)^\gamma} - \beta_i = 0\\ \alpha_i\left(1 - \frac{1}{\tilde{\tau}(1-\tilde{p}_i)^\gamma}\eta_i - y_i\left(w^\top\phi(x_i)+b\right)\right) &= 0\\ \beta_i\eta_i &= 0\\ y_i(w^\top\phi(x_i)+b) &\geq 1 - \frac{1}{\tilde{\tau}(1-\tilde{p}_i)^\gamma}\eta_i\\ \eta_i &\geq 0\\ \alpha_i &\geq 0\\ \beta_i &\geq 0\\ (for\ all\ i &= 1,\ldots,N)\end{aligned}$$

Substituting the above KKT conditions into (24), we can get the dual QPP:

$$\begin{aligned}&\min_{\alpha}\ \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j\kappa(x_i,x_j) - \sum_{i=1}^{N}\alpha_i\\&s.t.\ \sum_{i=1}^{N}\alpha_i y_i = 0\\&\qquad 0 \leq \alpha_i \leq \tilde{\tau}(1-\tilde{p}_i)^\gamma C, \quad i=1,\ldots,N\end{aligned} \tag{25}$$

Solve the dual problem (25) by QP slover, SMO [36], etc [37], [38] to obtain the optimum point α^* and the separating function and decision function can be represented as:

$$f(x) = \sum_{i=1}^{N} y_i\alpha_i^*\kappa(x,x_i) + b^* \tag{26}$$

$$h(x) = sgn(f(x)) \tag{27}$$

where b^* is determined by the KKT conditions. In practice, b^* is commonly gotten as

$$b^* = \frac{1}{N_{\{SV\}}}\sum_{i\in\{SV\}}\left(y_i - \sum_{j\in\{SV\}}\alpha_j y_j\kappa(x_i,x_j)\right) \tag{28}$$

where $\{SV\}$ denotes the set of support vectors and $N_{\{SV\}}$ is the number of support vectors.

Next, we introduce the proposed Adaptive FH-SVM in detail, which is an iterative algorithm. On account of the change of $\tilde{p}_i$ from a variable to a constant during the training SVM model, we update the values according to (15) after a $\hat{\text{FH}}$-SVM classifier is obtained and further consider the new values as the posteriori information applied to the next $\hat{\text{FH}}$-SVM model. In this way, the weight of sample loss can be updated automatically such that the similar effect to (21) can be achieved. Algorithm 1 presents the proposed Adaptive FH-SVM algorithm. It's obvious that the computational complexity is $O(TN^3)$, where T is the iterations and

N is the number of training samples. In addition, due to $\tilde{p}^s < \tilde{p}^{s+1}$(the superscript s denotes the s−th iteration), the objective function value of (23) satisfies $J(w^s, b^s, \tilde{p}^s) \geq J(w^{s+1}, b^{s+1}, \tilde{p}^s) \geq J(w^{s+1}, b^{s+1}, \tilde{p}^{s+1})$. And we know that $J(w, b, \tilde{p}) \geq 0$, i.e., the objective value is lower bounded, thus Algorithm 1 converges.

Algorithm 1 Adaptive FH-SVM Algorithm

Require:

-$S = \{(x_i, y_i)\}_{i=1}^N$: training set.
-C: penalty parameter.
-τ, γ: FH Loss parameters.
-$\kappa(\cdot,\cdot)$: kernel trick.
-it: number of iterative times.
-$itmax$: the maximum number of cycles.

Ensure:

The decision function: $h(x)$.

Initialize:
- $it = 0$.
- Initialize the scaling factor by:
1)Get the separating function $f(x)$ by solving C-SVM (2) with S;
2)Calculate $\{p_i\}_{i=1}^N$ by (15)(19);
3)Obtain the initial scaling factor $\{\tilde{\tau}(1-\tilde{p}_i)^\gamma\}_{i=1}^N$.

while $it < itmax$ **do**
  Solve dual QPP (25) with S and $\tilde{\tau}(1-\tilde{p}_i)^\gamma$ $(i = 1, \ldots, N)$;
  Obtain $f(x)$ and compute $f(x_i)$ $(i = 1, \ldots, N)$ by (26);
  Compute p_i $(i = 1, \ldots, N)$ by (15);
  Obtain $\tilde{p}_i$ $(i = 1, \ldots, N)$ by (19);
  Update the scaling factor $\tilde{\tau}(1-\tilde{p}_i)^\gamma$ $(i = 1, \ldots, N)$;
  it=it+1;
end while
return $h(x) = sgn(f(x))$.

IV. EXPERIMENTS

In this section, we validate the effectiveness of Adaptive FH-SVM for binary classification on 31 real-world imbalanced datasets. Meanwhile, compare our algorithm with other five state-of-the-art methods.

A. EXPERIMENTAL SETUP

All experiments are implemented by MATLAB R2012b on a PC with Intel Core I5 processor (3.10GHz) with 4 GB RAM and 64-bits operating system. In addition, in order to search for the best parameters, we take advantage of fivefold cross validation [28] for each technique on each dataset. And each experiments is repeated five times to report the average and standard deviation of each evaluation measures. Moreover, for simple calculation, we omit the bias term b and conduct the following augmentation: $w^\top \leftarrow (w^\top, b), x^\top \leftarrow (x^\top, 1)$. Furthermore, we adopt the QP solver in MATLAB toolbox to solve the dual QPP for our algorithm. Other settings are stated as follows.

TABLE 1. Description on *KEEL* datasets.

Dataset	Data size	Features	Imbalance ratio
Glass1	214	9	1.82
Pima	768	8	1.87
Glass0	214	9	2.06
Haberman	306	3	2.78
Vehicle1	846	18	2.90
Glass0123vs456	214	9	3.20
Ecoli1	336	7	3.36
Newthyroid2	215	5	5.14
Ecoli2	336	7	5.46
Glass6	214	9	6.38
Ecoli3	336	7	8.60
Yeast05679vs4	528	8	9.35
Vowel0	988	13	9.98
Glass016vs2	192	9	10.29
Glass2	214	9	11.59
Shuttle-c0vsc4	1829	9	13.87
Yeast1vs7	459	8	14.30
Glass4	214	9	15.47
Ecoli4	336	7	15.80
Pageblocks13vs4	472	10	15.86
Abalone9-18	731	8	16.40
Glass016vs5	472	9	19.44
Shuttle-c2vsc4	129	9	20.50
Yeast1458vs7	693	8	22.10
Glass5	214	9	22.78
Yeast2vs8	482	8	23.10
Yeast4	1484	8	28.10
Yeast1289vs7	947	8	30.57
Yeast5	1484	8	32.73
Ecoli0137vs26	281	7	39.14
Yeast6	1484	8	41.40

1) DATASETS

Here, we describe the adopted real-world imbalanced datasets. The benchmark datasets are from the *KEEL* dataset repository[1] [32], which has been widely used for class imbalance problem. To validate the effectiveness of the proposed method, we select 31 binary class datasets with imbalance ratio (IR) from 1.82 to 41.40, where

$$IR = \frac{number\ of\ positive\ instances}{number\ of\ negative\ instances}$$

The information of the selected datasets are summarized in Table 1. It should be pointed out that in order to maintain the imbalance ratio of the dataset, during cross validation, we divide each class into five parts and combine positive and negative instances from the same part into the final fivefold.

[1] Available at https://sci2s.ugr.es/keel/datasets.php

TABLE 2. Confusion matrix.

	Predicted positive	Predicted negative
Actual positive	TP*(True positive)*	FN*(False negative)*
Actual negative	FP*(False positive)*	TN*(True negative)*

2) BENCHMARK METHODS

- ROS-SVM: Random Over-sampling (ROS) is a basic sampling method, which duplicates the minority class instance randomly until the number of the minority reaches a given size. Then train a SVM model with the processed dataset.
- RUS-SVM: Random under-sampling (RUS) is also a common sampling technique, which eliminates a fraction of the majority class instances randomly. Then a SVM classifier is obtained by the processed majority and the original minority.
- SMOTE-SVM: Synthetic Minority Over Sampling Technique (SMOTE) [9] is a more strong over-sampling approach implementing before training SVM models. Instead of replicating the existing instance directly, it generates new synthetic instances by the k-nearest neighhors (kNN) of minority instances.
- WSVM: Weighted SVM has been introduced in previous section. In experiments, we set the values of C_+ and C_- in (6) as $N/(2 \cdot N_{pos})$ and $N/(2 \cdot N_{neg})$ respectively, where N is the total number of instances, N_{pos} is that of positive class and N_{neg} is that of negative class. On this occasion, C_+/C_- is equal to the ratio of the majority to minority class, which can obtain the reasonably good performance reported at [6]. It should be point out that we employ LIBSVM[1] [29] to implement WSVM in this literature.
- IRUS-SVM: Inverse Random Under-sampling (IRUS) [39] is an combination of the ensemble approach and the under-sampling method. Instead of classic random under-sampling to the majority class, IRUS highly under-samples the majority class such that the ratio of two classes is reversed to the original ratio. According to the modified sampling method and the bagging approach, the false positive rate can be relatively decreased. Similar to the above sampling methods, IRUS also adopts SVM to the basis classifier in this paper, in order to the fair comparison.

3) EVALUATION MEASURES

In this paper, we view *G-mean* [30] and *F-measure* [31] as evaluation measures to assess the performance of methods, which are both based on the confusion matrix as shown in Table 2.

G-mean is a typical assessment criterion for imbalanced datasets and is defined as:

$$G-mean = \sqrt{sensitivity \times specificity}$$

[1] Available at https://www.csie.ntu.edu.tw/cjlin/libsvm/

where *sensitivity* is true positive rate and *specificity* is true negative rate, described by the following equations respectively:

$$sensitivity = \frac{TP}{TP + FN}$$
$$specificity = \frac{TN}{TN + FP}$$

Thus, *G-mean* is the geometric mean of sensitivity and specificity. *F-measure* is represented as:

$$F-measure = \frac{(1+\beta^2) * sensitivity * precision}{\beta^2 * sensitivity + precision}$$

where

$$precision = \frac{TP}{TP + FP}$$

and β adjusts the importance of *sensitivity* and *precision*. In this paper, we set $\beta = 1$, that is to say, each has the same importance.

4) KERNELS

For all the techniques, we apply the Gaussian radial basis function (RBF)

$$\kappa(x, x') = \exp(- \| x - x' \|^2 /2\epsilon^2)$$

on the all datasets. The choice set of the parameter ϵ is introduced next.

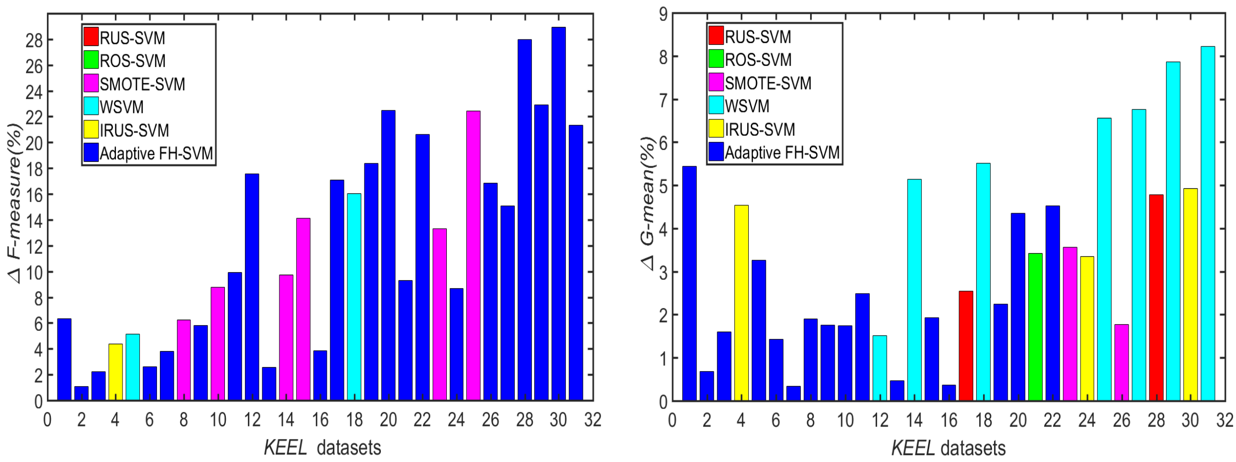
5) PARAMETERS

To obtain the best parameters for each approach, we use fivefold cross validation. For the fair comparison, the penalty parameter C and the kernel parameter ϵ are selected from the range of $\{10^{-3}, 10^{-2}, \ldots, 10^2, 10^3\}$ and $\{2^{-5}, 2^{-4}, \ldots, 2^4, 2^5\}$ respectively, in whichever methods they are. Meanwhile, the parameters τ and γ in the proposed method are tuned among the values $\{0.05, 0.1, 0.2, 0.25, 0.3, 0.4, 0.5, 0.6, 0.75, 0.8, 0.9, 0.95\}$ and $\{0, 0.1, 0.2, 0.5, 1, 2, 3, 5, 10\}$, respectively. Taking the optimization complexity into consideration, we set the maximum number of iterations as a default value 10 at the experiment part.

In addition, the imbalance ratio (IR) after adjustment is set to 1 for all pre-processing methods except for IRUS-SVM. In terms of IRUS-SVM, the parameter determining the quantity of the selected majority instances and the number of classifiers is selected from $\{0.05, 0.10, \ldots, 0.95\}$, as is shown in [39].

B. EXPERIMENTAL RESULTS

In this section, we compare the performance of Adaptive FH-SVM with all the benchmark methods. Table 3-4 list the results of all six algorithms on all 31 *KEEL* datasets, about *F-measure* and *G-mean* respectively. To be easy to check, these two tables report the detailed information of each dataset under its name, including the dimension (Dim), IR and the dataset's size. Furthermore, under each *F-measure*



(a) (b)

FIGURE 2. Plots representing the improved value between the *F-measure/G-mean* of the winning method and the mean *F-measure/G-mean* of other five methods on each *KEEL* dataset. And the color indicates the winning method. (a) Δ*F-measure*. (b) Δ*G-mean*.

or *G-mean* value, there are the relative standard deviation and the score. The value of score is in the range of 1 to 6. The smaller the score, the better the algorithm performs on the corresponding dataset. On the last three rows, there are the average rank, i.e. the mean score on all datasets, final rank and the statistic results *Win/Draw/Loss* for each algorithm.

According to the information of Table 3 and Table 4, we can draw the following conclusions. First, Adaptive FH-SVM outperforms other methods on more than half of the datasets, no matter which evaluation criteria is used. As shown in tables, Adaptive FH-SVM obtains the lowest average rank and the most counts of wins with respect to both *F-measure* and *G-mean*. Second, in some datasets, the values of *F-measure* or *G-mean* for Adaptive FH-SVM algorithm are close to the best, although our method is slightly worse.

To further compare the performance of all six methods, Figure 2(a) shows the improved value between the *F-measure* of the winning method and the mean *F-measure* of other five methods on each dataset while Figure 2(b) shows the improved *G-mean* where the color indicates the winning method on this dataset. As can be seen from this figures, our proposed method improves the values of evaluation criteria to some different extend on many datasets. It should be pointed out that on some datasets (e.g. *Pageblocks13vs4*, *Ecoli0137vs26*) the *F-measure* value is promoted over 20% because of the poor performance of IRUS-SVM, as shown in Table 3. To our surprise, the *G-mean* values of IRUS-SVM on the same datasets are comparative to that of other methods. The phenomenon indicates the model excessively focuses on the positive class such that the accuracy of the negative class is impaired. For example, there is an imbalanced dataset including 20 positive instances and 1000 negative instances. A model classifies all positive instances correctly and 100 negative instances are misclassified, which can obtain a high *G-mean* 94.87% and a low *F-measure* 16.67%. As shown in Figure 2, the higher the imbalance ratio, the more obvious the situation. Because with the decrease of the positive class' scale, *precision* is more easily determined by the negative class. In the one hand, this phenomenon caters to the starting point of our paper, that is, the accuracy of each class is important. In the other hand, this phenomenon demonstrates the necessity of two evaluation criteria. Only if both measures are high can the algorithm be effective. Both high *F-measure* and *G-mean* value on most datasets verify the validity of our propose Adaptive FH-SVM.

To further measure the similarity among the used algorithms, we utilize the nonparametric statistical test, named Friedmen test [40]. In the test, H_0-hypothesis implies that there is no significant difference among all algorithms. If the value F_F is larger than the value q_α, the above hypothesis is rejected. F_F is defined as:

$$\chi_F^2 = \frac{12N}{k(k+1)}\Big[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\Big] \tag{29}$$

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2} \tag{30}$$

where the average rank of the jth algorithm $R_j = \frac{1}{N}\sum_i r_i^j$, and r_i^j is the rank of the jth of k algorithms on the ith of N datasets. The equation to calculate q_α is given as:

$$q_\alpha = F\big(\alpha, (k-1), (k-1)(N-1)\big) \tag{31}$$

That is, q_α is obtained from the F-distribution with $\big((k-1), (k-1)(N-1)\big)$ degree of freedom and the level of significance α. Without loss of generality, we select the *F-measure* values to apply the Friedman Test and the corresponding test results are listed in Table 5. As is shown in the table, $F_F > q_\alpha$ holds when $\alpha = 0.05$, $\alpha = 0.025$ and $\alpha = 0.01$, which implies that there are indeed significance differences among six algorithms.

TABLE 3. ***F-measure*** **(%), Standard Deviation and Score of five state-of-the-art methods and the proposed method on all 31 selected** ***KEEL*** **datasets.**

Datasets *Name* *Dim/IR/Size*	RUS-SVM F-measure Std/Score	ROS-SVM F-measure Std/Score	SMOTE-SVM F-measure Std/Score	WSVM F-measure Std/Score	IRUS-SVM F-measure Std/Score	Adaptive FH-SVM F-measure Std/Score
Glass1	69.48	63.97	70.01	69.63	71.92	75.33
9 × 1/1.82/214	2.27/5	2.00/6	1.12/3	1.71/4	0.98/2	1.87/1
Pima	67.73	67.60	68.95	68.05	69.33	69.41
8 × 1/1.87/768	0.43/5	0.47/6	0.67/3	0.74/4	0.42/2	0.53/1
Glass0	75.75	75.67	77.67	76.85	77.84	78.96
9 × 1/2.06/214	1.24/5	1.70/6	1.19/3	1.87/4	0.91/2	1.65/1
Haberman	50.60	49.99	50.52	49.51	54.85	51.68
3 × 1/2.78/306	1.99/3	0.67/5	0.80/4	1.22/6	0.80/1	1.59/2
Vehicle1	69.74	72.41	72.56	76.84	67.86	75.86
18 × 1/2.90/846	0.91/6	0.73/4	0.57/3	1.36/1	0.39/5	0.75/2
Glass0123vs456	87.45	88.81	90.51	90.62	87.87	91.64
9 × 1/3.20/214	0.79/6	0.97/4	1.31/3	0.49/2	0.67/5	0.96/1
Ecoli1	78.11	77.89	80.65	78.29	73.95	81.59
7 × 1/3.36/336	0.67/4	0.53/5	1.22/2	0.26/3	0.76/6	1.28/1
Newthyroid2	97.65	98.40	98.67	97.59	70.65	97.90
5 × 1/5.14/215	1.35/3	0.53/2	0.00/1	1.06/4	0.69/6	0.63/3
Ecoli2	86.26	7.85	88.08	87.53	69.80	89.72
7 × 1/5.46/336	0.38/5	0.83/3	0.38/2	0.42/4	0.60/6	0.83/1
Glass6	85.86	85.72	90.35	86.03	68.34	82.02
9 × 1/6.38/214	1.85/3	0.74/4	1.56/1	3.33/2	2.21/6	3.04/5
Ecoli3	62.10	64.39	66.64	64.54	56.37	72.71
7 × 1/8.60/336	1.65/5	2.36/4	0.96/2	1.99/3	2.27/6	2.21/1
Yeast05679vs4	46.37	46.80	49.52	48.46	37.54	63.28
8 × 1/9.35/528	0.48/5	1.17/4	3.54/2	2.34/3	1.69/6	1.12/1
Vowel0	97.03	100.00	100.00	100.00	90.09	100.00
13 × 1/9.98/988	0.84/5	0.00/1	0.00/1	0.00/1	1.28/6	0.00/1
Glass016vs2	37.90	50.35	53.73	49.07	34.41	48.28
9 × 1/10.29/192	2.65/5	4.32/2	3.66/1	1.63/3	0.78/6	2.33/4
Glass2	38.28	52.58	59.94	50.56	34.40	53.26
9 × 1/11.59/214	2.08/5	1.66/3	3.74/1	5.53/4	3.20/6	5.64/2
Shuttle-c0vsc4	99.84	99.84	99.92	99.92	80.92	99.92
9 × 1/13.87/1829	0.19/4	0.19/4	0.16/1	0.16/1	0.56/6	0.16/1
Yeast1vs7	33.92	33.49	33.42	32.62	24.16	48.60
8 × 1/14.30/459	2.16/2	0.96/3	1.81/4	1.23/5	1.73/6	6.35/1
Glass4	63.32	77.65	76.89	81.83	34.37	76.70
9 × 1/15.47/214	6.84/5	7.56/2	2.87/3	3.59/1	0.41/6	4.12/4
Ecoli4	81.55	80.82	83.68	80.57	33.98	90.49
7 × 1/15.80/336	4.17/3	2.92/4	1.03/2	5.02/5	0.93/6	1.44/1
Pageblocks13vs4	71.35	92.00	98.89	79.48	41.11	99.02
10 × 1/15.86/472	3.36/5	4.07/3	1.56/2	7.36/4	1.38/6	1.33/1
Abalone9-18	36.86	47.15	45.80	46.20	23.08	49.10
8 × 1/16.40/731	1.87/5	0.63/2	0.78/4	1.78/3	0.44/6	1.63/1
Glass016vs5	60.31	83.47	86.40	84.42	31.63	89.87
9 × 1/19.44/472	9.76/5	5.24/4	4.86/2	4.35/3	1.63/6	1.07/1
Shuttle-c2vsc4	98.67	100.00	100.00	88.00	63.04	83.73
9 × 1/20.5/129	2.67/3	0.00/1	0.00/1	6.53/4	6.10/6	8.14/5
Yeast1458vs7	15.12	19.06	18.39	17.63	17.43	26.20
8 × 1/22.10/693	0.94/6	2.76/2	2.45/3	2.19/4	2.14/5	3.80/1
Glass5	58.10	85.35	88.80	85.09	26.83	76.55
9 × 1/22.78/214	6.52/5	6.04/2	5.36/1	3.58/3	0.82/6	3.72/4
Yeast2vs8	42.07	59.71	53.61	66.52	35.54	68.32
8 × 1/23.10/482	8.15/5	2.21/3	5.08/4	2.17/2	2.98/6	3.99/1
Yeast4	28.17	32.14	30.78	30.58	15.28	42.45
8 × 1/28.10/1484	0.82/5	2.18/2	1.55/3	0.53/4	0.24/6	1.43/1
Yeast1289vs7	16.38	17.71	17.88	17.93	13.04	44.55
8 × 1/30.57/947	1.44/5	1.70/4	1.56/3	0.74/2	0.90/1	2.61/6
Yeast5	52.76	66.98	67.35	55.35	16.37	74.65
8 × 1/32.73/1484	2.28/5	2.66/3	4.21/2	1.33/4	0.07/6	2.07/1
Ecoli0137vs26	48.50	46.18	55.60	64.40	20.80	76.00
7 × 1/39.14/281	9.03/4	4.22/5	6.87/3	7.52/2	2.29/6	3.27/1
Yeast6	44.88	42.46	47.61	35.19	12.90	57.92
8 × 1/41.40/1484	2.12/3	1.65/4	4.53/2	4.29/5	0.44/6	1.11/1
Average Rank	**4.5161**	**3.4516**	**2.3226**	**3.2258**	**5.3548**	**1.7097**
Final Rank	**5**	**4**	**2**	**3**	**6**	**1**
Win/Draw/Loss	**28/1/2**	**24/1/6**	**22/2/7**	**23/2/6**	**30/0/1**	**-/-/-**

TABLE 4. *G-mean* (%), Standard Deviation and Score of five state-of-the-art methods and the proposed method on all 31 selected *KEEL* datasets.

Datasets *Name* *Dim/IR/Size*	RUS-SVM G-mean Std/Score	ROS-SVM G-mean Std/Score	SMOTE-SVM G-mean Std/Score	WSVM G-mean Std/Score	IRUS-SVM G-mean Std/Score	Adaptive FH-SVM G-mean Std/Score
Glass1	75.75	71.08	76.17	76.02	77.78	80.80
9 × 1/1.82/214	2.07/5	1.39/6	0.83/3	1.18/4	0.88/2	1.42/1
Pima	74.92	74.92	76.00	75.23	76.02	76.10
8 × 1/1.87/768	0.37/5	0.38/5	0.54/3	0.61/4	0.43/2	0.63/1
Glass0	82.52	82.00	83.99	83.39	84.29	84.83
9 × 1/2.06/214	1.09/5	1.42/6	0.87/3	1.38/4	0.96/2	1.28/1
Haberman	64.62	64.34	64.30	63.79	69.12	65.84
3 × 1/2.78/306	1.86/3	0.68/4	0.54/5	0.80/6	0.88/1	1.54/2
Vehicle1	82.98	84.68	84.83	87.39	81.99	87.63
18 × 1/2.90/846	0.84/5	0.60/4	0.46/3	0.77/2	0.34/6	0.80/1
Glass0123vs456	94.40	93.57	95.77	95.48	94.94	96.26
9 × 1/3.20/214	0.32/6	0.68/2	0.73/5	0.51/1	0.21/3	0.49/4
Ecoli1	89.79	89.72	89.81	89.82	88.72	89.91
7 × 1/3.36/336	0.87/4	0.34/5	0.51/3	0.70/2	0.51/6	1.05/1
Newthyroid2	99.25	99.66	99.72	99.25	91.24	99.72
5 × 1/5.14/215	0.48/4	0.11/3	0.00/1	0.66/4	0.28/6	0.14/1
Ecoli2	92.29	93.63	93.86	94.15	89.87	94.51
7 × 1/5.46/336	1.18/5	0.49/4	0.91/3	0.39/2	0.36/6	0.29/1
Glass6	93.20	92.53	93.73	92.93	90.87	94.39
9 × 1/6.38/214	0.81/3	3.02/5	0.33/2	0.70/4	0.77/6	0.55/1
Ecoli3	89.40	88.74	87.47	89.85	86.93	90.97
7 × 1/8.60/336	1.25/3	1.04/4	0.86/5	0.73/2	0.59/6	0.46/1
Yeast05679vs4	78.06	77.89	79.26	79.75	77.96	78.05
8 × 1/9.35/528	4.09/3	0.84/6	1.32/2	0.76/1	0.45/5	0.86/4
Vowel0	98.97	100.00	100.00	100.00	98.73	100.00
13 × 1/9.98/988	1.025	0.00/1	0.00/1	0.00/1	0.10/6	0.00/1
Glass016vs2	73.25	76.78	75.47	80.78	76.65	76.06
9 × 1/10.29/192	6.14/6	2.55/2	5.00/5	5.09/1	1.29/3	5.69/4
Glass2	78.98	76.93	80.25	80.72	77.10	80.72
9 × 1/11.59/214	3.82/4	6.96/6	3.51/3	4.69/1	1.66/5	5.25/1
Shuttle-c0vsc4	99.99	99.99	99.99	99.92	98.26	99.99
9 × 1/13.87/1829	0.01/1	0.01/1	0.01/1	0.16/5	0.06/6	0.01/1
Yeast1vs7	76.65	75.20	74.34	76.24	72.46	72.26
8 × 1/14.30/459	2.18/1	3.57/3	3.61/4	0.87/2	0.52/5	6.59/6
Glass4	93.83	84.92	88.75	94.80	86.70	92.26
9 × 1/15.47/214	1.75/2	7.24/6	4.70/4	1.01/1	0.22/5	1.66/3
Ecoli4	92.49	90.33	92.78	93.11	86.61	93.31
7 × 1/15.80/336	1.43/4	1.18/5	1.26/3	3.54/2	0.48/6	0.96/1
Pageblocks13vs4	96.88	95.57	99.24	97.25	88.99	99.93
10 × 1/15.86/472	0.14/4	0.69/5	1.46/2	0.81/3	1.96/6	0.09/1
Abalone9-18	81.19	85.51	85.22	85.04	76.54	82.48
8 × 1/16.40/731	2.73/5	0.72/1	0.83/2	1.03/3	0.61/6	1.40/4
Glass016vs5	93.87	97.81	95.89	98.84	87.85	99.37
9 × 1/19.44/472	0.95/5	2.39/3	2.89/4	0.32/2	0.58/6	0.12/1
Shuttle-c2vsc4	99.92	100.00	100.00	88.49	95.12	98.67
9 × 1/20.50/129	0.16/3	0.00/1	0.00/1	6.93/6	2.18/5	0.67/4
Yeast1458vs7	66.79	62.19	66.81	65.91	68.47	63.89
8 × 1/22.10/693	1.26/3	3.75/6	2.68/2	1.48/4	1.69/1	4.90/5
Glass5	86.59	93.59	98.20	99.11	87.09	97.32
9 × 1/22.78/214	6.89/6	3.77/4	2.16/2	0.12/1	0.33/5	1.93/3
Yeast2vs8	75.89	73.50	76.68	73.97	76.66	74.56
8 × 1/23.10/482	1.44/3	3.67/6	3.01/2	4.64/5	2.29/1	2.76/4
Yeast4	78.26	76.00	79.23	84.44	77.38	77.51
8 × 1/28.10/1484	10.57/3	6.78/6	7.64/2	0.34/1	0.86/5	1.58/4
Yeast1289vs7	72.16	61.90	71.28	69.18	71.35	63.19
8 × 1/30.57/947	3.24/1	8.97/6	4.22/3	5.87/4	1.34/2	1.26/5
Yeast5	87.78	89.91	92.87	97.44	82.86	94.47
8 × 1/32.73/1484	5.86/5	1.58/4	0.99/3	0.13/1	0.08/6	1.90/2
Ecoli0137vs26	80.15	76.14	71.12	73.77	80.46	76.49
7 × 1/39.14/281	7.42/2	18.22/4	10.76/6	6.85/5	1.61/1	2.87/3
Yeast6	81.62	81.20	77.85	89.05	80.27	83.22
8 × 1/41.40/1484	2.85/3	1.41/4	5.57/6	0.44/1	1.07/5	0.21/2
Average Rank	**3.7419**	**4.2581**	**2.9032**	**2.8065**	**4.4516**	**2.2903**
Final Rank	**4**	**6**	**3**	**2**	**5**	**1**
Win/Draw/Loss	**21/1/9**	**25/2/4**	**19/3/9**	**18/2/11**	**24/0/7**	**-/-/-**

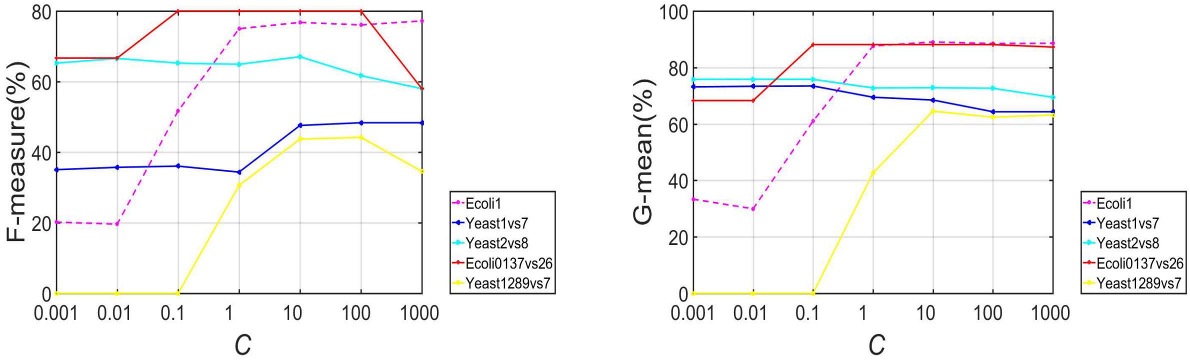


FIGURE 3. ***F-measure*** **values (%) and** ***G-mean*** **values (%) of Adaptive FH-SVM with respect to** C **on the used datasets.**

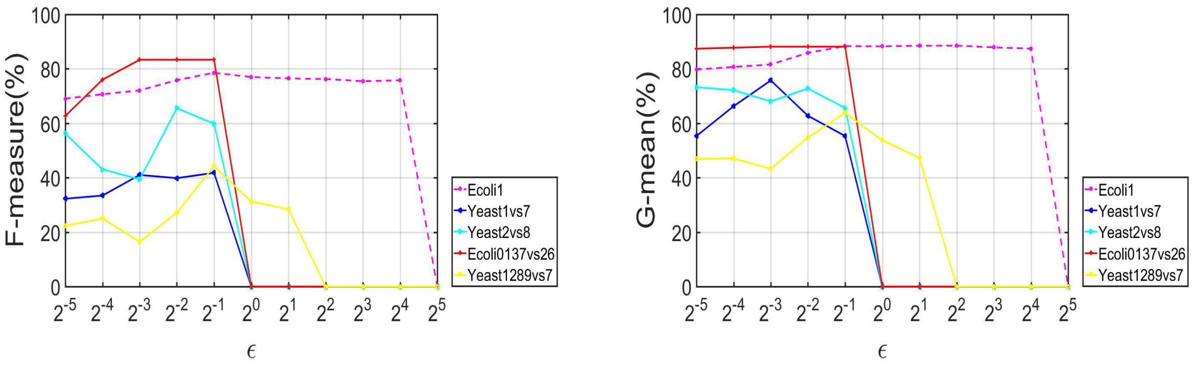


FIGURE 4. ***F-measure*** **values (%) and** ***G-mean*** **values (%) of Adaptive FH-SVM with respect to** ϵ **on the used datasets.**

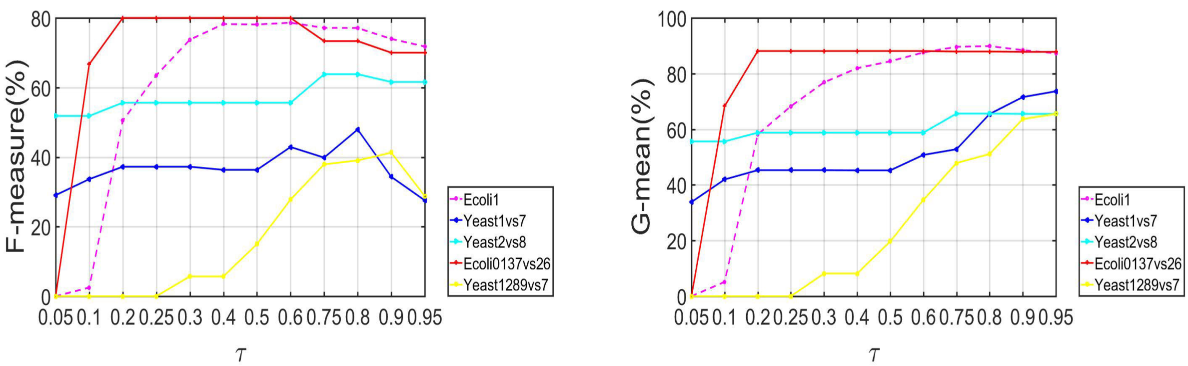


FIGURE 5. ***F-measure*** **values (%) and** ***G-mean*** **values (%) of Adaptive FH-SVM with respect to** τ **on the used datasets.**

C. ANALYSIS OF PARAMETERS

In the proposed Adaptive FH-SVM, there are four parameters, i.e. the regularization parameter C, the Gaussian kernel parameter ϵ, parameters τ, γ in the scaling factor, impacting the classification performance. Therefore, we discuss the influence of these four parameters to our proposed algorithm on 5 datasets selected from Table 1, that is, *Ecoli1*, *Yeast1vs7*, *Yeast2vs8*, *Ecoli0137vs26*, *Yeast1289vs7*. It should be declared that when we discuss one parameter, other parameters are set to the best ones selected by fivefold cross validation.

Figure 3-6 respectively show the *F-measure* value and *G-mean* value of our proposed method with respect to $\{C, \epsilon, \tau, \gamma\}$ on the above datasets. According to the figures, the following conclusions can be found: (1) with the increase of C, the performance take on the growth trend or steady state. Overall speaking, a relatively small C is not proper to the high performance. (2) With the increase of ϵ, the values take on a trend of first fluctuation and then stability. And when ϵ exceeds 0.5, the classification performance suffers a significant decrease on most datasets, which demonstrates that a small ϵ should be concentrated on during tuning

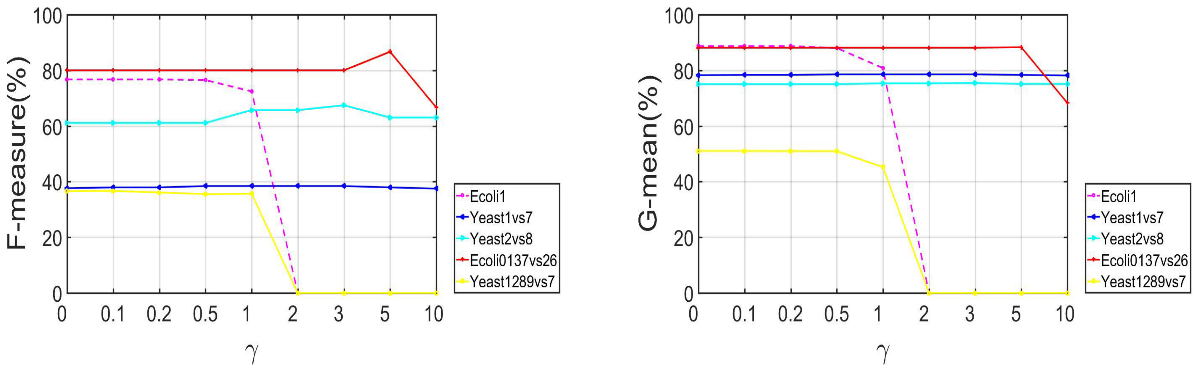


FIGURE 6. ***F-measure*** **values (%) and** ***G-mean*** **values (%) of Adaptive FH-SVM with respect to γ on the used datasets.**

TABLE 5. **Results of Friedmen test.**

α	χ_F^2	F_F	q_α	H_0-hypothesis
0.05	54.9667	16.4845	2.2745	reject
0.025	54.9667	16.4845	2.6521	reject
0.01	54.9667	16.4845	3.1416	reject

parameters. (3) The value of τ has different influence on different datasets. For example, τ has more effect on *Yeast1289vs7* than *Yeast2vs8*. But generally speaking, a large value is more beneficial to the performance. (4) With the change of γ, the performance changes are relatively smooth and a small value makes more sense.

V. CONCLUSION

In this paper, we reshaped Hinge Loss with Focal Loss, and further proposed the FH Loss devoting to the imbalanced binary classification. FH loss aims at improving the accuracy of each class and achieving a better trade-off between the positive and negative class performance so that the influence of imbalance can be reduced. In order to solve the corresponding optimization problem easily, we modified the FH loss and proposed the Adaptive FH-SVM algorithm. The effectiveness of our method is verified by comparing with five state-of-the-art methods on 31 imbalanced datasets. Furthermore, at the end of the experimental part, we analysed the influence of parameters on the classification performance.

But without ignorance, due to the unsuitability for SVM on large-scale datasets, the proposed method is confronted with the same problem. Besides, the resource-constrained devices and the selection of QP solver enlarge the training difficulty on large-scale datasets. In the future, we will explore faster algorithm to solve large-scale problems with the proposed loss function and further extend to multiclass classification.

REFERENCES

[1] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2008.

[2] P. Li, K. L. Chan, and S. Fu, "Kernel machines for imbalanced data problem in biomedical applications," in *Support Vector Machines Applications*. Cham, Switzerland: Springer, 2014, pp. 221–268.

[3] Z. Zheng, X. Wu, and R. Srihari, "Feature selection for text categorization on imbalanced data," *ACM SIGKDD Explor. Newslett.*, vol. 6, no. 1, pp. 80–89, 2004.

[4] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intell. Data Anal.*, vol. 6, no. 5, pp. 429–449, Oct. 2002.

[5] C. Cortes and V. Vapnik, "Support vector machine," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[6] R. Akbani, S. Kwek, and N. Japkowicz, "Applying support vector machines to imbalanced datasets," in *Proc. Eur. Conf. Mach. Learn.* Berlin, Germany: Springer, 2004, pp. 39–50.

[7] G. Wu and E. Y. Chang, "KBA: Kernel boundary alignment considering imbalanced data distribution," *IEEE Trans. Knowl. Data Eng.*, vol. 7, no. 6, pp. 786–795, Jun. 2005.

[8] R. Batuwita and V. Palade, "Class imbalance learning methods for support vector machines," in *Imbalanced Learning: Foundations, Algorithms, and Applications*. 2013.

[9] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, no. 1, pp. 321–357, 2002.

[10] S. Hu, Y. Liang, L. Ma, and Y. He, "MSMOTE: Improving classification performance when training data is imbalanced," in *Proc. 2nd Int. Workshop Comput. Sci. Eng.*, vol. 2, 2009, pp. 13–17.

[11] S. Barua, M. M. Islam, X. Yao, and K. Murase, "MWMOTE–majority weighted minority oversampling technique for imbalanced data set learning," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 2, pp. 405–425, Feb. 2013.

[12] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "SMOTEBoost: Improving prediction of the minority class in boosting," in *Proc. Eur. Conf. Princ. Data Mining Knowl. Discovery*. Berlin, Germany: Springer, 2003, pp. 107–119.

[13] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 1, pp. 185–197, Jan. 2010.

[14] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 2, pp. 539–550, Apr. 2009.

[15] K. Veropoulos, C. Campbell, and N. Cristianini, "Controlling the sensitivity of support vector machines," in *Proc. Int. Joint Conf. AI*, vol. 55, Jul. 1999, p. 60.

[16] T. Imam, K. M. Ting, and J. Kamruzzaman, "z-SVM: An SVM for improved classification of imbalanced data," in *Proc. Australas. Joint Conf. Artif. Intell.* Berlin, Germany: Springer, 2006, pp. 264–273.

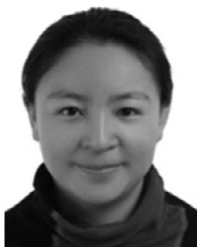
[17] R. Batuwita and V. Palade, "FSVM-CIL: Fuzzy support vector machines for class imbalance learning," *IEEE Trans. Fuzzy Syst.*, vol. 18, no. 3, pp. 558–571, Jun. 2010.

[18] C.-F. Lin and S.-D. Wang, "Fuzzy support vector machines," *IEEE Trans. Neural Netw.*, vol. 13, no. 2, pp. 464–471, Mar. 2002.

[19] D. M. J. Tax, ''One-class classification: Concept learning in the absence of counter-examples,'' Tech. Rep., 2002.
[20] B. Schölkopf, J. C. Platt, and J. Shawe-Taylor, ''Estimating the support of a high-dimensional distribution,'' *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, Jul. 2001.
[21] B. Raskutti, '' Extreme re-balancing for SVMS: A case study,'' *ACM Sigkdd Explor. Newslett.*, vol. 6, no. 1, pp. 60–69, Jun. 2004.
[22] A. Kowalczyk and B. Raskutti, ''One class SVM for yeast regulation prediction,'' *ACM SIGKDD Explor. Newslett.*, vol. 4, no. 2, pp. 99–100, Dec. 2002.
[23] J. Shawe-Taylor, P. L. Bartlett, R. C. Williamson, and M. Anthony, ''Structural risk minimization over data-dependent hierarchies,'' *IEEE Trans. Inf. Theory*, vol. 44, no. 5, pp. 1926–1940, Sep. 1998.
[24] T. Y. Lin, P. Goyal, and R. Girshick, ''Focal loss for dense object detection,'' in *Proc. IEEE Int. Conf. Comput. Vis.*, Jul. 2017, pp. 2980–2988.
[25] V. N. Vapnik, ''An overview of statistical learning theory,'' *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 988–999, Sep. 1999.
[26] L. Devroye, L. Györfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*, vol. 31. Springer, 2013.
[27] X. Huang, L. Shi, and J. A. K. Suykens, ''Support vector machine classifier with pinball loss,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 5, pp. 984–997, May 2014.
[28] U. M. Braga-Neto and E. R. Dougherty, ''Is cross-validation valid for small-sample microarray classification,'' *Bioinformatics*, vol. 20, no. 3, pp. 374–380, Feb. 2004.
[29] C.-C. Chang and C.-J. Lin, ''LIBSVM: A library for support vector machines,'' *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, Apr. 2011, Art. no. 27.
[30] M. Kubat and S. Matwin, ''Addressing the curse of imbalanced training sets: One-sided selection,'' in*Proc. ICML*, vol. 97, Jul. 1997, pp. 179–186.
[31] A. Estabrooks and N. Japkowicz, ''A mixture-of-experts framework for learning from imbalanced data sets,'' in *Proc. Int. Symp. Intell. Data Anal.* Berlin, Germany: Springer, 2001, pp. 34–43.
[32] J. Alcalá-Fdez, A. Fernúndez, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, ''Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework,'' *J. Multiple-Valued Logic Soft Comput.*, vol. 17, pp. 255–287, Jun. 2011.
[33] G.-L. Qian, S.-N. Gu, and J.-S. Jiang, ''The dynamic behaviour and crack detection of a beam with a crack,'' *J. Sound Vib.*, vol. 138, no. 2, pp. 233–243, Apr. 1990.
[34] L. Zhang, F. Yang, Y. D. Zhang, and Y. J. Zhu, ''Road crack detection using deep convolutional neural network,'' in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 3708–3712.
[35] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.
[36] J. Platt, ''Sequential minimal optimization: A fast algorithm for training support vector machines,'' Tech. Rep., 1998.
[37] E. Osuna, R. Freund, and F. Girosi, ''An improved training algorithm for support vector machines,'' in *Proc. Neural Netw. Signal Process., IEEE Signal Process. Soc. Workshop*, Sep. 1997, pp. 276–285.
[38] L. Zanni, T. Serafini, and G. Zanghirati, ''Parallel software for training large scale support vector machines on multiprocessor systems,'' *J. Mach. Learn. Res.*, vol. 7, pp. 1467–1492, Jul. 2006.
[39] M. A. Tahir, J. Kittler, and F. Yan, ''Inverse random under sampling for class imbalance problem and its application to multi-label classification,'' *Pattern Recognit.*, vol. 45, no. 10, pp. 3738–3750, 2012.
[40] J. Demšar, ''Statistical comparisons of classifiers over multiple data sets,'' *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

QI WANG received the bachelor's degree from the College of Mathematics and Systems Science, Shandong University of Science and Technology, Shandong, China, in 2017. She is currently pursuing the master's degree with the School of Mathematical Science, University of Chinese Academy of Sciences, Beijing, China. Meanwhile, she is studying in the Research Center on Fictitious Economy and Data Science, Chinese Academy of Sciences. Her research interests include optimization, machine learning, and data mining.

YINGJIE TIAN received the bachelor's degree in mathematics, in 1994, the master's degree in applied mathematics, in 1997, and the Ph.D. degree in management science and engineering. He is currently a Professor with the Research Center on Fictitious Economy and Data Science, Chinese Academy of Sciences. He has published four books on SVMs. His research interests include support vector machines, optimization theory and applications, data mining, intelligent knowledge management, and risk management.



DALIAN LIU is currently an Associate Professor with Beijing Union University. Her current research interests include optimization and data mining.